

[0020]

Embodiment

The embodiment of the present invention will be described below.

Embodiment 1: Fig. 1 shows a constitution of a network including a computer system of a first embodiment of the present invention. Numeral 1 shows a client computer for executing a program. Numeral 2 shows a server computer for transferring the program to the client computer 1. Numeral 3 shows a network for connecting the client computer 1 and the server computer 2. Numeral 4 shows a program loading and executing means, for loading and executing a program, which is provided in the client computer 1. Numeral 5 shows a program transferring means, for transferring a program, which is provided in the server computer 2. Numeral 6 shows a cache control means for controlling caches 8, 11, which is included in the program loading and executing means 4. Numeral 7 shows a non-volatile storage apparatus including the caches 8, 11, which is connected to the client computer 1. Numeral 8 shows a conventionally used normal cache (a first cache) which is provided in the non-volatile storage apparatus 7. Numeral 9 shows a non-volatile storage apparatus connected to the server computer 2. Numeral 10 shows a program stored in the non-volatile storage apparatus 9. Numeral 11 shows a permanent cache (a second cache) for storing data until a user deletes it. The client computer 1 and the non-volatile storage apparatus 7 constitute a computer system of the present invention.

[0021]

Fig. 2 shows a flowchart of operations of the program

loading and executing means 4. The operations of the program loading and executing means 4 will be discussed below with reference to the flowchart shown in Fig. 2.

[0022]

If the user specifies the program to be stored in the permanent cache 11 and inputs a starting order for storing it in the computer system, the program loading and executing means 4 firstly determines whether the program file is stored in the permanent cache 11, by accessing the cache controlling means 6 (step ST1). If the program file is stored therein, the program file is loaded using the cache controlling means 6 (step ST2). If the program file is not stored therein, the program loading and executing means 4 also determines whether the program file is stored in the normal cache 8, by accessing the cache controlling means 6 (step ST3).

[0023]

If the program file is stored in the normal cache 8, the program file is loaded from the normal cache 8 (step ST5), and if the program file is not stored therein, the program file is downloaded, using the program transferring means 5, through the network 3 (step ST4).

[0024]

When the program file is loaded through the normal cache 8 or the program transferring means 5, the program file is checked to determine whether it has been specified, by the user, to be stored in the permanent cache or is a file which has been loaded due to a need for the program file specified by the user to be stored (step ST6). If the program file is specified, by the user, to be stored or the program file is a file which is necessary for the program file specified, by the user, to be stored, the program file

is stored in the permanent cache 11 through the cache controlling means 6 (step ST 7), and if the program file is not specified, by the user, to be stored or the program file is not a file which is necessary for the program file specified, by the user, to be stored, the program file is stored in the normal cache 8 (step ST8). When the program file is stored in the normal cache 8, a countermeasure for overflow of cache is conducted in accordance with the cache controlling policy of the cache controlling means 6. For example, a file, which has not been used for the longest time and is the oldest file among those stored in the cache, is deleted from the normal cache 8, and a new file is stored in the cache.

[0025]

After file loading and the cache process are finished, the program is executed and a dependency relationship analysis is performed, so as to analyze the relationship between the dependent and to-be-dependent of the file which is necessary for the program file when executing (step ST9). If the file, which is dynamically necessary and is to be loaded for the dependency relationship analysis and execution of the file (step ST10), does not exist, the cache process ends. Otherwise, the process returns back to the cache check and the loading step (step ST1).

[0026]

As mentioned above, according to the first embodiment, if the program file cannot be downloaded through the program transferring means 5, in other words, the client computer 1 is for a mobile use, etc., and the program is not stored in the normal cache 8 and the program cannot be loaded from the normal cache 8, the effect for executing the program can be obtained by loading the program file from the permanent cache 11 because the program file

specified by the user and the file which is necessary for the specified program file are stored in the permanent cache 11.

If the program is stored in the permanent cache 11, whether the program file which is specified to be stored by the user or the file which is necessary for the program file is determined. Therefore, only the program file which the user wishes to be stored is stored in the permanent cache 11, and the effect that the cache capacity cannot be redundantly consumed can be obtained. Furthermore, if the program is provided as modules and constituted by a plurality of files, the user can store the necessary file groups without specifying the name of the program precisely, using the dependency relationship analysis function of the program loading and executing means. To update the file groups in the permanent cache 11, similar steps, in the storing process, should be conducted.

DS

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平10-301831

(43)公開日 平成10年(1998)11月13日

(51)Int.Cl.⁶
G 0 6 F 12/00

識別記号
5 4 7
5 1 4

F I
G 0 6 F 12/00

5 4 7 A
5 1 4 A

審査請求 未請求 請求項の数9 O L (全 20 頁)

(21)出願番号 特願平9-105011

(22)出願日 平成9年(1997)4月22日

(71)出願人 000006013

三菱電機株式会社

東京都千代田区丸の内二丁目2番3号

(72)発明者 岡田 英明

東京都千代田区丸の内二丁目2番3号 三

菱電機株式会社内

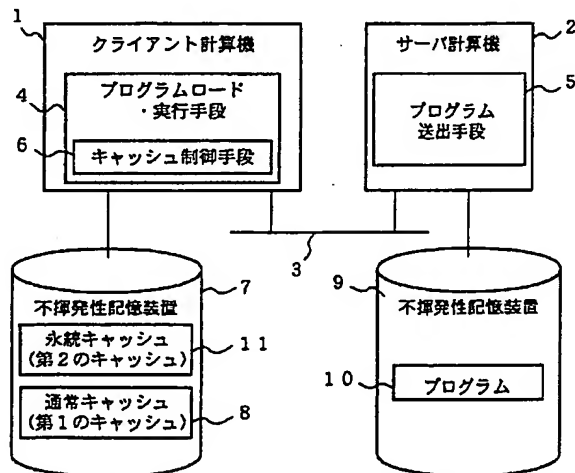
(74)代理人 弁理士 田澤 博昭 (外1名)

(54)【発明の名称】 計算機システム

(57)【要約】

【課題】 キャッシュが溢れ、プログラムの内容がキャッシュから消去されてしまうような場合、プログラムを再ロードできない場合がある。

【解決手段】 プログラムファイルを通常キャッシュに一時的に保存すると共に、永続キャッシュを不揮発性記憶装置中に設け、少なくともユーザの指定するプログラムファイルを保存し、ユーザの消去の指示があるまでのプログラムファイルを保存し続ける。



【特許請求の範囲】

【請求項1】 プログラムファイルを外部から取り込み、取り込んだプログラムを実行するクライアント計算機と、該クライアント計算機が前記プログラムを実行するために必要なファイルを記録する不揮発性記憶装置とを有する計算機システムにおいて、前記プログラムファイルを一時的に保存する第1のキャッシュと、前記不揮発性記憶装置中に設けられ、少なくともユーザの指定するプログラムファイルを保存し、ユーザの消去の指示があるまで該プログラムファイルを保存し続ける第2のキャッシュとを備えたことを特徴とする計算機システム。

【請求項2】 第1のキャッシュ中へのプログラムファイルの保存の事実と、保存されたプログラムファイルの依存関係とを記録するキャッシュ履歴簿を不揮発性記憶装置中に設けたことを特徴とする請求項1記載の計算機システム。

【請求項3】 プログラムファイルの依存関係解析機能を持ち、プログラムをロード、キャッシュするプログラムロード・キャッシュ手段をクライアント計算機に設けたことを特徴とする請求項1または請求項2記載の計算機システム。

【請求項4】 ユーザから削除するように指示のあったファイル群を第2のキャッシュから削除し、第1のキャッシュへコピーするキャッシュ移動手段をクライアント計算機に設けたことを特徴とする請求項1から請求項3のうちのいずれか1項記載の計算機システム。

【請求項5】 第2のキャッシュ中のファイル群の削除履歴簿を不揮発性記憶装置中に設けたことを特徴とする請求項4記載の計算機システム。

【請求項6】 複数のサーバに存在するディレクトリが異なる同一ファイルのファイル名を所定のディレクトリのファイル名に一致化させるパス一致化手段をクライアント計算機に設けたことを特徴とする請求項1から請求項5のうちのいずれか1項記載の計算機システム。

【請求項7】 第2のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを前記第2のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックする更新チェック手段をクライアント計算機に設けたことを特徴とする請求項1から請求項6のうちのいずれか1項記載の計算機システム。

【請求項8】 第2のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを前記第2のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックし、更新されていた場合、さらにそのファイルの依存関係解析を行なってロードする更新チェック・ロード手段をクライアント計算機に設けたことを

特徴とする請求項1から請求項6のうちのいずれか1項記載の計算機システム。

【請求項9】 プログラムに使用期限がある場合に、ユーザの指定した使用期限と実際の使用頻度にあわせて使用期限が近付いていることの警告を行う使用期限チェック手段をクライアント計算機に設けたことを特徴とする請求項1から請求項8のうちのいずれか1項記載の計算機システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 この発明は計算機システムに係り、特にネットワークからプログラムをダウンロードして実行するのに適した計算機システムに関するものである。

【0002】

【従来の技術】 図22は従来の計算機システムを組み込んだネットワークの構成図であり、図において、21はプログラムを実行するクライアント計算機、22はプログラムをクライアント計算機21に送り出すサーバ計算機、23はクライアント計算機21とサーバ計算機22とを接続するネットワーク、24はクライアント計算機21に設けられた、プログラムをロードして実行するプログラムロード・実行手段、25はサーバ計算機22に設けられた、プログラムを送出するプログラム送出手段、26はプログラムロード・実行手段24に含まれ、キャッシュ28を制御するキャッシュ制御手段、27はクライアント計算機21に接続され、キャッシュ27を含む不揮発性記憶装置、28は不揮発性記憶装置27上のキャッシュ、29はサーバ計算機22に接続されている不揮発性記憶装置、30は不揮発性記憶装置29上に保存されているプログラムである。クライアント計算機21と不揮発性記憶装置27とは計算機システムを構成する。

【0003】 次に動作について説明する。クライアント計算機21は、ハイパーテキスト言語 (Hyper Text Markup Language (以下、「HTML」と略記する)) を使用したWWW (World Wide Web) 等のネットワーク23上で、プログラムロード・実行手段24により、Java言語などの言語で記述された実行可能なプログラムをサーバ計算機22からダウンロードして実行する。

【0004】 すなわち、まず、プログラムロード・実行手段24は、ネットワーク23を介して、サーバ計算機22のプログラム送出手段25にアクセスしてプログラム送出手段25内に保持されているHTMLで記載されたHTML文書を開覧する。開覧したHTML文書は不揮発性記憶装置27内のキャッシュ28に保存される (後に読み出すためにキャッシュ中に一時的にプログラム、データ等を記憶する動作を以後「キャッシュする」と表現する)。このとき開覧したHTML文書にプログ

ラム30の実行が指示されていれば、プログラムロード・実行手段24はプログラム送出手段25にプログラム30の送出を依頼して、プログラム30をダウンロードし、実行する。通常、プログラム30もキャッシュ28中にキャッシュされ、次回にアクセスするときにはこのキャッシュ28中に保持されているプログラム30が利用される。プログラム30がサーバ計算機22により更新されたときに、更新されたプログラム30をダウンロードしたい場合には、クライアント計算機21はプログラムロード・実行手段24に再読み込みを指示する。このようにすることにより、ユーザはHTML文書を閲覧するのに、クライアント計算機21上にプログラム30を保存せずに必要なときのみサーバ計算機22からダウンロードすればよく、サーバ計算機22によるプログラム30の一元管理が可能となる。

【0005】

【発明が解決しようとする課題】従来の計算機システムは以上のように構成されているので、キャッシュ28がHTML文書やプログラムなどの他のデータにより溢れ、プログラム30の内容がキャッシュ28から消去されてしまうことがある。このような場合、再びプログラムロード・実行手段24がプログラム30を利用しようとすると、サーバ計算機22のプログラム送出手段25にプログラム30の送出を依頼する必要がある。このことはクライアント計算機21とサーバ計算機22とがネットワーク23により常に接続されている場合や、必要な時に接続可能であり、かつクライアント計算機21とサーバ計算機22とがプログラム30をダウンロードするのに十分な性能を持っている場合には問題ないが、携帯用のクライアント計算機21を外部に持ち出している場合などクライアント計算機21とサーバ計算機22との接続が困難でクライアント計算機21側の性能も十分でないような場合では、プログラム30を利用できないという課題があった。また、プログラムファイルがCD-ROM等に記録されている場合にも、CD-ROMを携帯しないとそのプログラムを実行できないという課題があった。

【0006】そのような携帯用のクライアント計算機21が不揮発性記憶装置27上にキャッシュ28のみでなく、ユーザが操作可能なファイルシステムを備えている場合には、プログラム30をそのファイルシステム上に保存しておき、ネットワーク23にクライアント計算機21が接続されていない外部でも実行することが可能ではあるが、サーバ計算機22上のプログラム30が更新された場合、ユーザが自ら再びそれをダウンロードし、保存しなければならず、サーバ計算機22によるプログラム30の一元管理の可能性などの利点を損なってしまいうという課題もある。

【0007】また、プログラム30が複数のファイルから構成されている場合、プログラムロード・実行手段2

4はプログラム30中に記録されている情報をもとに自動的に必要なファイルの更なるロード・実行を行うが、常にユーザにとって必要なファイルが正確にロード・実行されるとは限らず、ユーザが、ファイル間の支配・被支配関係である依存関係を判断して複数のファイルをロード・保存する必要がある場合もあるという課題もあった。

【0008】この発明は上記のような課題を解決するためになされたもので、携帯用のクライアント計算機などネットワーク接続が常には行えないようなクライアント計算機端末においても、ネットワーク上のプログラムやCD-ROM等の記録媒体上のプログラムを常に行うことができるようにクライアント計算機に保存でき、ユーザが簡単にプログラムの保存や更新を行うことのできる計算機システムを得ることを目的とする。

【0009】また、通常のキャッシュが溢れた場合にもプログラムが勝手に削除されることのない計算機システムを得ることを目的とする。

【0010】さらに、ユーザが依存関係をいちいちチェックしなくとも、依存関係にあるファイルを自動的にキャッシュできる計算機システムを得ることを目的とする。

【0011】

【課題を解決するための手段】請求項1記載の発明に係る計算機システムは、プログラムファイルを一時的に保存する第1のキャッシュと、不揮発性記憶装置中に設けられ、少なくともユーザの指定するプログラムを保存し、ユーザの消去の指示があるまでそのプログラムを保存し続ける第2のキャッシュとを備えたものである。

【0012】請求項2記載の発明に係る計算機システムは、第1のキャッシュ中へのプログラムファイルの保存の事実と、保存されたプログラムファイルの依存関係とを記録するキャッシュ履歴簿を不揮発性記憶装置中に設けたものである。

【0013】請求項3記載の発明に係る計算機システムは、プログラムファイルの依存関係解析機能を持ち、プログラムをロード、キャッシュするプログラムロード・キャッシュ手段をクライアント計算機に設けたものである。

【0014】請求項4記載の発明に係る計算機システムは、ユーザから削除するように指示のあったファイル群を第2のキャッシュから削除し、第1のキャッシュへコピーするキャッシュ移動手段をクライアント計算機に設けたものである。

【0015】請求項5記載の発明に係る計算機システムは、第2のキャッシュ中のファイル群の削除履歴簿を不揮発性記憶装置中に設けたものである。

【0016】請求項6記載の発明に係る計算機システムは、複数のサーバに存在するディレクトリが異なる同一ファイルのファイル名を所定のディレクトリのファイル

名に一致化させるバス一致化手段をクライアント計算機に設けたものである。

【0017】請求項7記載の発明に係る計算機システムは、第2のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを第2のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックする更新チェック手段をクライアント計算機に設けたものである。

【0018】請求項8記載の発明に係る計算機システムは、第2のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを第2のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックし、更新されていた場合、さらにそのファイルの依存関係解析を行なってロードする更新チェック・ロード手段をクライアント計算機に設けたものである。

【0019】請求項9記載の発明に係る計算機システムは、プログラムに使用期限がある場合に、ユーザの指定した使用期限と実際の使用頻度にあわせて使用期限が近づいていることの警告を行う使用期限チェック手段をクライアント計算機に設けたものである。

【0020】

【発明の実施の形態】以下、この発明の実施の一形態を説明する。

実施の形態1. 図1はこの発明の実施の形態1の計算機システムを組み込んだネットワークを示す構成図である。図において、1はプログラムを実行するクライアント計算機、2はプログラムをクライアント計算機1に送り出すサーバ計算機、3はクライアント計算機1とサーバ計算機2とを接続するネットワーク、4はクライアント計算機1に設けられた、プログラムをロードして実行するプログラムロード・実行手段、5はサーバ計算機2に設けられた、プログラムを送出するプログラム送出手段、6はプログラムロード・実行手段4に含まれ、キャッシュ8、11を制御するキャッシュ制御手段、7はクライアント計算機1に接続され、キャッシュ8、11を含む不揮発性記憶装置、8は不揮発性記憶装置7上の従来から用いられている通常キャッシュ(第1のキャッシュ)、9はサーバ計算機2に接続されている不揮発性記憶装置、10は不揮発性記憶装置9上に保存されているプログラム、11はユーザの消去の指示があるまでデータを保存し続けることのできる永続キャッシュ(第2のキャッシュ)である。クライアント計算機1と不揮発性記憶装置7とは、この発明の計算機システムを構成する。

【0021】図2はプログラムロード・実行手段4の動作を示すフローチャートである。以下、図2のフローチャートを参照しながらプログラムロード・実行手段4の動作を説明する。

【0022】ユーザが永続キャッシュ11へ保存するプログラムを指定し、その保存の実行開始命令を計算機システムに入力すると、プログラムロード・実行手段4は、まず、そのプログラムファイルが永続キャッシュ11にキャッシュされているか、キャッシュ制御手段6を介して問い合わせる(ステップST1)。キャッシュされている場合、キャッシュ制御手段6を介してそのプログラムファイルをロードする(ステップST2)。キャッシュされていない場合、再びキャッシュ制御手段6を介してそのプログラムファイルが通常キャッシュ8にキャッシュされているか否かを問い合わせる(ステップST3)。

【0023】そのプログラムファイルが通常キャッシュ8にキャッシュされている場合、通常キャッシュ8よりプログラムファイルをロードし(ステップST5)、キャッシュされていない場合、ネットワーク3を介してプログラム送出手段5よりプログラムファイルをダウンロードする(ステップST4)。

【0024】通常キャッシュ8またはプログラム送出手段5よりプログラムファイルがロードされると、それがユーザが永続キャッシュに保存するように指定したプログラムファイル、またはユーザが保存するように指定したプログラムファイルが必要としたためロードすることとなったファイルであるか否かをチェックする(ステップST6)。保存指定されたプログラムファイルまたは保存指定されたプログラムファイルが必要としたファイルである場合、キャッシュ制御手段6を介し永続キャッシュ11に保存し(ステップST7)、保存指定されたプログラムファイルまたは保存指定されたプログラムファイルが必要としたファイル以外のファイルである場合には、通常キャッシュ8にキャッシュする(ステップST8)。通常キャッシュ8にキャッシュする場合にはキャッシュ制御手段6のキャッシュ制御方針に従ってキャッシュ溢れ対策が行なわれる。例えば、最も使われておらず、最も古くキャッシュされたファイルを通常キャッシュ8から削除した上で新たなファイルをキャッシュする。

【0025】ファイルのロードおよびキャッシュ処理が終了すると、プログラムを実行すると共に、その実行の際にそのプログラムファイルが必要とするファイルの相互の支配・被支配関係を解析する依存関係解析を行う(ステップST9)。依存関係解析および実行に伴って動的に必要とされロードする必要のあるファイルが存在しない場合(ステップST10)、キャッシュ処理は終了し、そうでない場合には、キャッシュのチェックおよびロード(ステップST1)に戻る。

【0026】以上のように、この実施の形態1によれば、プログラムファイルをプログラム送出手段5よりダウンロードすることができない場合、すなわちクライアント計算機1が携帯用である等の場合には、通常キャッ

シユ8の中にプログラムがキャッシュされておらず通常キャッシュ8からロードできなくとも、ユーザの指定したプログラムファイル及び指定したプログラムファイルが必要としたファイルについては永続キャッシュ11中に保存されているため、永続キャッシュ11からそのプログラムファイルをロードすることによりプログラムを実行することができる効果が得られる。また永続キャッシュ11にプログラムを保存するにあたって、ユーザが保存指定したプログラムファイルまたはそのプログラムファイルが必要とするファイルであるか否かを判定するため、永続キャッシュ11には本当に保存しておきたいプログラムファイルのみ保存され、キャッシュ領域を無駄に消費しないという効果が得られる。さらにプログラムがモジュール化されており数多くのファイルにより構成される場合でも、プログラムロード・実行手段の依存関係解析機能によりユーザがファイル名を詳細に指定しなくとも、必要なプログラムファイル群を保存できる効果が得られる。なお、永続キャッシュ11中のプログラムファイル群を更新したい場合には、保存処理と同様の処理を行う。

【0027】実施の形態2. 図3はこの発明の実施の形態2の計算機システムを組み込んだネットワークを示す構成図である。図3において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0028】図3において、1aはクライアント計算機、7aは不揮発性記憶装置、12はプログラムファイルを通常キャッシュ8にキャッシュする場合に依存関係解析の結果を記録するキャッシュ履歴簿、13は通常キャッシュ8から永続キャッシュ11にプログラムファイルをコピーするキャッシュコピー手段である。図4はキャッシュ履歴簿12の記憶内容を示す図であり、ファイル名と依存関係とを記録してある。図4の例では、ユーザの指示によりファイル名main. classのファイルが記録され、ファイルmain. classを実行するのに必要なファイルとしてファイル名sub. classのファイルがファイルmain. classにより指示されたことを表している。クライアント計算機1aと不揮発性記憶装置7aとはこの発明の計算機システムを構成する。

【0029】次に動作について説明する。図5はこの実施の形態2のプログラムロード・実行手段4の動作を示すフローチャートであり、図6はキャッシュコピー手段13の動作を示すフローチャートである。図5において、図2に示す実施の形態1のプログラムロード・実行手段4の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【0030】この実施の形態2においては、ステップST9でプログラムを実行して依存関係解析を行った後、保存指定されたプログラムファイル及びそのプログラム

ファイルが必要とするファイル以外のファイルを通常キャッシュ8にキャッシュした(ステップST8)場合、ステップST9における依存関係解析の結果をキャッシュ履歴簿12に記録する(ステップST11)。すなわち、永続キャッシュ11に保存すべきファイルを指定していない場合には、永続キャッシュ11への保存は行なわれない。また、複数のファイルから構成されるプログラムファイル群においては、永続キャッシュ11への保存を途中から指定すれば、指定したファイル以降のファイルについては永続キャッシュ11への保存が可能になるが、それではプログラムの開始地点となるプログラムファイルが保存されていないことになり、意味をなさない。このような場合に、通常キャッシュ8の大きさやキャッシュ制御手段6の制御方針にもよるが、永続キャッシュ11に保存されなかったプログラムファイルも通常キャッシュ8にはキャッシュされている可能性が高い。そこで、通常キャッシュ8へのキャッシュ動作の依存関係解析結果をキャッシュ履歴簿12に記録し、通常キャッシュ8から永続キャッシュ11へのプログラムファイルのコピーに利用するのである。

【0031】このキャッシュ履歴簿12には、図4に示すように、プログラムファイルのファイル名と共に、ステップST9のプログラム実行、依存関係解析の結果が付記されている。したがって、プログラム実行途中に永続キャッシュ11への保存を指定した場合、このキャッシュ履歴簿12に記録されている依存関係を逆にたどればプログラムの開始地点を特定でき、この開始地点のプログラムファイルから始まり、順にキャッシュ履歴簿12をたどってプログラムファイルのロード、永続キャッシュ11への保存を行えば、保存を指示した地点までのファイルが保存され、また、それ以降に必要なファイルは実施の形態1と同様にして保存できる。また、プログラム実行中であるため、永続キャッシュ11に保存されなかったプログラムファイルも通常キャッシュ8にキャッシュされている可能性は非常に高いため、通常キャッシュ8から永続キャッシュ11へ移動させれば、ネットワークを使うオーバーヘッドが減り、永続キャッシュ11や通常キャッシュ8の有効的な利用となる。さらに、通常キャッシュ8にプログラムが残っていれば、永続キャッシュ11への移動が可能となるため、一旦ネットワークからの接続を切ってしまった場合でも永続キャッシュ11への保存が可能となる。

【0032】次に、キャッシュコピー手段13による通常キャッシュ8から永続キャッシュ11へのプログラムファイルのコピー動作を図6のフローチャートを参照しながら説明する。ユーザが、あるプログラムファイルを通常キャッシュ8から永続キャッシュ11へコピーすることを指定すると、キャッシュコピー手段13はキャッシュする先頭のファイルを探索する。すなわち、まず、最初のコピーターゲットファイルをユーザの指定したフ

ファイルとする(ステップST21)。

【0033】次に、キャッシュ履歴簿12を検索し、コピーターゲットファイルの読み込みを指示した履歴の記録があるか否かを判定する(ステップST22)。検索の結果コピーターゲットファイルの読み込みを指示した履歴の記録がキャッシュ履歴簿12中に存在しない場合には、キャッシュコピー手段13はプログラムファイルのコピーを諦めて動作を終了する。読み込みを指示した履歴の記録が存在する場合には、この読み込みを指示したファイルを新しいコピーターゲットファイルとする(ステップST23)。

【0034】次に、この新しいコピーターゲットファイルが依存関係の先頭にあるか否かを判断し(ステップST24)、先頭になければ、ステップST22～ST24の動作を繰り返し、依存関係の先頭にあるコピーターゲットファイル(すなわち、ユーザの指示したファイル(図4の例ではファイルmain.class))を求める。

【0035】依存関係の先頭にあるファイルが求められたら、このファイルを先頭として通常キャッシュ8から永続キャッシュ11にコピーする(ステップST25)。

【0036】以上のように、この実施の形態2によれば、永続キャッシュ11に保存し損なったプログラムも、ネットワーク3を介さずに通常キャッシュ8からコピーでき、ネットワーク3を用いるオーバーヘッドを減らすことができるという効果が得られる。

【0037】実施の形態3。図7はこの発明の実施の形態3の計算機システムを組み込んだネットワークを示す構成図である。図7において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0038】図7において、1bはクライアント計算機、14はプログラムロード・実行手段4と同様なプログラム依存関係解析機能を持ち、プログラムをロード・キャッシュするプログラムロード・キャッシュ手段である。クライアント計算機1b及び不揮発性記憶装置7はこの発明の計算機システムを構成する。

【0039】実施の形態1の計算機システムにおいては、プログラムが複数のファイルで構成されている場合には、2番目以降のファイルはプログラムを実行した後に永続キャッシュ11にキャッシュされることとなる

(図2のステップST6、ST9、ST10参照)。これは、実施の形態1においては、プログラムロード・実行手段4は実行時のロード機能のみ提供していれば十分であるからであるが、プログラムを実行しないでもプログラムファイル群を永続キャッシュ11に保存しておきたい場合もある。この実施の形態3のプログラムロード・キャッシュ手段14はこのような要望に応えるものである。

【0040】次に動作について、図8のプログラムロード・キャッシュ手段14の動作を示すフローチャートを参照しながら説明する。プログラムロード・キャッシュ手段14はユーザより保存するプログラムファイルを指定されると、プログラム送出手段5に依頼してそのファイルをネットワーク3を介してダウンロードする(ステップST31)。

【0041】次に、ダウンロードしたプログラムファイルをキャッシュ制御手段6を介して永続キャッシュ11にキャッシュする(ステップST32)。

【0042】さらに、キャッシュしたプログラムファイルの依存関係解析を行ない(ステップST33)、この依存関係解析の結果更にロードすべきファイルがあるか否かを判定する(ステップST34)。この判定の結果、更にロードすべきファイルが存在する場合には、ステップST31に戻ってファイルのロードを行う。ロードすべきファイルが存在しない場合には動作を終了する。

【0043】以上のように、この実施の形態3によれば、ユーザはプログラムを実行することなくプログラムファイル群を保存できるだけでなく、プログラムファイル群の構成によってはすべての機能を実行しなければすべてのファイル群を保存することができなかったという実施の形態1の問題点を克服することができる効果が得られる。

【0044】実施の形態4。図9はこの発明の実施の形態4の計算機システムを組み込んだネットワークを示す構成図である。図9において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0045】図9において、1cはクライアント計算機、15はユーザが指定したファイル群を永続キャッシュ11から通常キャッシュ8へ移動するキャッシュ移動手段である。クライアント計算機1c及び不揮発性記憶装置7はこの発明の計算機システムを構成する。

【0046】実施の形態1の計算機システムにおいては、永続キャッシュ11へのプログラムファイルの保存が永続キャッシュ溢れにより不可能となった場合には、キャッシュをあきらめるか、通常キャッシュ8にキャッシュするしかない。そこで実施の形態4では、永続キャッシュ11が溢れた場合、永続キャッシュ11から通常キャッシュ8へユーザの指定したプログラムファイルを移動させるキャッシュ移動手段15を設けたのである。

【0047】次に動作について説明する。図10はこの実施の形態4のプログラムロード・実行手段4の動作を示すフローチャートであり、図11はキャッシュ移動手段15の動作を示すフローチャートである。図10において、図2に示す実施の形態1のプログラム・ロード実行手段4の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【0048】この実施の形態4においては、ステップS T7でキャッシュ制御手段6を介して永続キャッシュ11にプログラムファイル及びそのプログラムファイルが必要とするファイルをキャッシュした後、永続キャッシュ11が溢れてキャッシュに失敗したか否かを判定し（ステップS T12）、キャッシュに成功していればそのままプログラムを実行して依存関係解析を行う（ステップS T9）。永続キャッシュ11が溢れてキャッシュに失敗した場合には、キャッシュ移動手段15のサブルーチン呼び出して実行した（ステップS T13）後、プログラムを実行して依存関係解析を行う（ステップS T9）。

【0049】キャッシュ移動手段15は、まず、永続キャッシュ11からどのファイル群を通常キャッシュ8に移動するかをユーザに問い合わせる（ステップS T41）。これは永続キャッシュ11の記憶内容を勝手に削除しないという永続キャッシュの特徴から避けて通れないステップである。問い合わせは、永続キャッシュ11にキャッシュされているすべてのファイルについてではなく、ファイル群の単位で問い合わせれば十分である。ユーザが移動すべきファイル群を指示すると、そのファイル群を永続キャッシュ11から通常キャッシュ8へ移動する（ステップS T42）。

【0050】なお、永続キャッシュ11に永続キャッシュ11から削除して通常キャッシュ8に移動したファイルの履歴を記録した削除履歴簿を設けてもよい。この削除履歴簿を参照することにより、永続キャッシュ11中の削除すべきファイルのヒントをユーザに与えることができ、効率の良い永続キャッシュを実現することが可能となる。

【0051】以上のように、この実施の形態4によれば、永続キャッシュ11を効率良く利用でき、さらに通常キャッシュ8へ移動されたファイル群の可用性については保証はされないが、この保存を行ってから携帯用計算機システムを持って外出する場合など、運用方法によっては通常キャッシュ8もうまく使用することにより永続キャッシュ11から削除されたプログラムも使用することができる効果が得られる。

【0052】実施の形態5。図12はこの発明の実施の形態5の計算機システムを組み込んだネットワークを示す構成図である。図12において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0053】図12において、1dはクライアント計算機、4aはプログラムロード・実行手段、6aはキャッシュ制御手段、16は複数のサーバに存在するディレクトリが異なる同一ファイルのファイル名を所定のディレクトリのファイル名に一致化させるバス一致化手段である。

【0054】図13に示すように、計算機Host Aに

はディレクトリ/program/scheduleの下にプログラムを構成する4つのファイルが格納されており、計算機Host Bにはディレクトリ/service/scheduleの下に同様に4つのファイルが格納されており、例えば計算機Host Aは事業所Aに、計算機Host Bは事業所Bにあり、携帯用計算機システムを持つ営業マンが、ときには事業所Aを、ときには事業所Bを訪れるものとする。実施の形態1の場合、事業所AにおいてプログラムファイルHost A: /program/schedule/main.classをプログラム開始ファイルとして永続キャッシュ11に保存した場合、事業所Bで同じプログラム群を更新、保存しようとするればファイルのパスが異なるため事業所Aで保存したのと同じ内容のファイル群が永続キャッシュ11中に別に保存されることとなり、永続キャッシュ11が無駄に消費される。これを避けようとするれば同じプログラムが両方の事業所において提供されているにもかかわらず、営業マンは意識して一方のみを使用しなければならない。そこで、この実施の形態5においては、キャッシュ制御手段6aにバス一致化手段16を追加したものである。

【0055】次に動作について説明する。図14はこの実施の形態5のプログラムロード・実行手段4aの動作を示すフローチャートである。図14において、図2に示す実施の形態1のプログラムロード・実行手段4の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【0056】この実施の形態5のプログラムロード・実行手段4aは、ユーザが永続キャッシュ11へ保存するプログラムを指定し、その保存の実行開始命令を計算機システムに入力すると、バス一致化手段16により、ユーザにより指定されたファイル名のパス名（ディレクトリ名）をBase1と変換する（ステップS T14）。すなわち、バス一致化手段16は、ユーザが入力したプログラムファイル名の開始ディレクトリが図13のファイル群1のBaseとして登録されているか否かを判断し、登録されている場合には入力された開始ディレクトリの如何にかかわらず、ファイル名の開始ディレクトリをHost A: /program/scheduleに変換して、以後の動作を行う。以後の動作は実施の形態1のプログラムロード・実行手段4の動作と同一である。

【0057】以上のように、この実施の形態5によれば、キャッシュ制御手段6aを介して永続キャッシュ11にキャッシュする（ステップS T7）か、または永続キャッシュ11にキャッシュされていてキャッシュ制御手段6aによってプログラムファイルをロードする場合（ステップS T2）に、図13のファイル群情報を利用して、プログラムロード・実行手段4aが欲しているファイルがHost A: /program/schedule

le/main.classであっても、HostB:/program/schedule/main.classであっても、ファイル群1中のmain.classを参照する。これにより複数のサーバを同様に扱うことができ、永続キャッシュ11を無駄にしない効果が得られる。

【0058】実施の形態6. 図15はこの発明の実施の形態6の計算機システムを組み込んだネットワークを示す構成図である。図15において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0059】図15において、1eはクライアント計算機、4bはプログラムロード・実行手段、6bはキャッシュ制御手段、17はプログラム送出手段5及び永続キャッシュ11から同一ファイル名のプログラムファイルをロードして内容を比較し、変更の有無を検出する、キャッシュ制御手段6bに設けられた更新チェック手段である。

【0060】次に動作について説明する。図16はこの実施の形態6のプログラムロード・実行手段4bの動作を示すフローチャートである。図16において、図2に示す実施の形態1のプログラム・ロード実行手段4の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【0061】この実施の形態6においては、永続キャッシュ11に必要なプログラムファイルが保存されていれば保存されているプログラムファイルをロードし(ステップST2)、計算機システムがネットワーク3に接続されているか否かを判断する(ステップST15)。計算機システムがネットワーク3に接続されていない場合は、ロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う(ステップST9)。計算機システムがネットワーク3に接続されているときには、更新チェック手段17がプログラム送出手段5から永続キャッシュ11に保存されているのと同じファイル名のプログラムファイルをダウンロードする(ステップST16)。

【0062】次に、更新チェック手段17はプログラム送出手段5からダウンロードしたプログラムファイルと永続キャッシュ11中のプログラムファイルとの内容を比較する(ステップST17)。その結果内容が相違している場合には変更があった旨を表示してユーザに知らせる(ステップST18)。以後、ネットワーク3に接続されていない場合と同様にダウンロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う(ステップST9)。

【0063】この場合、永続キャッシュ11の内容は勝手に更新しない。更新情報に基づいて更新したい場合には、ユーザはプログラム送出手段5から更新されたプログラムをダウンロードし(ステップST4)、キャッシュ

制御手段6bにより永続キャッシュ11(ステップST7)または通常キャッシュ8(ステップST8)にキャッシュすることにより更新する。この際、実施の形態2のように、通常キャッシュ8にキャッシュした場合に依存関係解析結果をキャッシュ履歴簿12(図3)に記録してもよい(図5のステップST11)。

【0064】以上のように、この実施の形態6によれば、容易に更新チェックができるだけでなく、タイムスタンプなどによる更新チェックではなくプログラムファイルの内容を比較するため、どの機能を司るファイルが更新されているか、大きく更新されているか少なく更新されているかなどの情報をユーザが知ることができる効果が得られる。また、残念ながらプログラムの更新により新たなバグが混入されたり、かえって使い勝手が悪くなってしまうことがよくある。このようなことによる影響を避けるため、永続キャッシュを変更することなく更新されたプログラムの動作チェックなどをすることができ効果が得られる。

【0065】実施の形態7. 図17はこの発明の実施の形態7の計算機システムを組み込んだネットワークを示す構成図である。図17において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0066】図17において、1fはクライアント計算機、4cはプログラムロード・実行手段、6cはキャッシュ制御手段、18はプログラム送出手段5及び永続キャッシュ11から同一ファイル名のプログラムファイルをロードして内容を比較し、変更の有無を検出し、更新されたファイルを永続キャッシュに保存して、更に更新されたファイルの依存関係解析を行って更新されたファイルにより必要とされるようになった新たなファイルをロード・保存する、キャッシュ制御手段6bに設けられた更新チェック・ロード手段である。

【0067】次に動作について説明する。図18はこの実施の形態7のプログラムロード・実行手段4cの動作を示すフローチャートである。図18において、図16に示す実施の形態6のプログラム・ロード実行手段4bの動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【0068】この実施の形態7においては、実施の形態6と同様に、永続キャッシュ11に必要なプログラムファイルが保存されていれば保存されているプログラムファイルをロードし(ステップST2)、計算機システムがネットワーク3に接続されているか否かを判断する(ステップST15)。計算機システムがネットワーク3に接続されていない場合はロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う(ステップST9)。計算機システムがネットワーク3に接続されているときには、更新チェック・ロード手段18がプログラム送出手段5から永続キャッシュ11に

保存されているのと同じファイル名のプログラムファイルをダウンロードする(ステップST16)。

【0069】次に、更新チェック手段17はプログラム送出手段5からダウンロードしたプログラムファイルと永続キャッシュ11中のプログラムファイルとの内容を比較する(ステップST17)。その結果内容が相違している場合には変更があった旨を表示してユーザに知らせる(ステップST18)。

【0070】この表示によりユーザがプログラムの更新を指示しなければ(ステップST19)、実施の形態6と同様に、ダウンロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う(ステップST9)。ユーザがプログラムの更新を指示した場合には(ステップST19)、更新チェック・ロード手段18はステップST17での比較結果により更新されているプログラムファイルを、実施の形態3のプログラムロード・キャッシュ手段14と同様に、ロードし(図8のステップST31参照)、永続キャッシュ11にキャッシュし(図8のステップST32参照)、そのプログラムファイルの依存解析を行って(図8のステップST33参照)プログラムファイルを更新する(図18のステップST20)。

【0071】以上のように、この実施の形態7によれば、永続キャッシュ11の更新が容易に、かつ更新をもれなく反映することができる効果が得られる。

【0072】実施の形態8。図19はこの発明の実施の形態8の計算機システムを組み込んだネットワークを示す構成図である。図19において、図1の実施の形態1の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【0073】図19において、1gはクライアント計算機、19はプログラムの使用期限をチェックする使用期限チェック手段、20は使用期限と使用の経歴を記録した使用期限記録簿である。

【0074】図20は不揮発性記憶装置7b中に設けられた使用期限記録簿20の記録内容の一例を示す図である。図20の例では、ファイル群Aには使用期限がなく、ファイル群Bは使用期限が1997年3月10日までであり、前回1997年1月25日に使用した実績があることを表している。

【0075】次に動作について説明する。試用あるいは課金のためなどによりプログラムの使用期限がある場合など、永続キャッシュ11中のプログラムが使用期限切れのため使用できず、永続キャッシュ11の効果が発揮できない場合がある。この実施の形態8の計算機システムは、例えば週報を作成するためのプログラムのようにユーザは週に一度程度しか使わないなどの情報とユーザが指定する使用期限とを勘案して、使用期限が近付いていることをユーザに警告するものである。

【0076】図21はこの実施の形態8のプログラムロ

ード・実行手段4の動作を示すフローチャートである。図21において、図2に示す実施の形態1のプログラム・ロード実行手段4の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【0077】この実施の形態8においては、キャッシュ制御手段6によりプログラムをロードした(ステップST2)後、次回の使用時期を予測するため、次回使用予想時期=今回使用時期-前回使用時期+今回使用時期を演算して求める(ステップST211)。

【0078】次に、このようにして求めた次回使用予想時期が使用期限を超えている場合には、その旨を表示してユーザに警告する(ステップST212)。

【0079】また、今回の使用時期が既に使用期限を超えているか否かを判断し(ステップST213)、使用期限を超えてしまっている場合には実行プロセスを終了する。使用期限を超えていない場合には、今回の使用時期を使用期限記録簿20に記し(ステップST213)、永続キャッシュ11にキャッシュされているプログラムを実行し、依存関係解析を行う(ステップST9)。

【0080】以上のように、この実施の形態8によれば、例えば月報を作成するためのプログラムならば週報プログラムの場合より使用期限と警告を出すべき時の間がより長くなる可能性が高いなどのプログラム使用頻度に応じた指摘を行うことができ、ユーザがプログラムの更新のためにそのプログラムのある事業所に出かけるなどの行動を取るための指針を与えることができる効果が得られる。

【0081】なお、以上のいずれの実施の形態においても、計算機システムはネットワーク中で用いる場合について説明したが、本発明の計算機システムはネットワーク中での使用に限定されるものではなく、プログラムファイルがCD-ROMやフロッピーディスク等の記録媒体に記録されていて、このような記録媒体等の外部のプログラムソースからプログラムファイルを取り込んで使用するような場合についても適用し得るものである。

【0082】

【発明の効果】以上のように、請求項1記載の発明によれば、不揮発性記憶装置中に永続キャッシュを設けるように構成したので、ネットワークに接続されていなくとも、またプログラムを記録した記録媒体が手元になくとも、プログラムが利用可能であるだけでなく、明示的に永続キャッシュを消去しないかぎりプログラムが利用可能であり、さらにプログラムが複数ファイルから構成される場合でもプログラムロード・実行手段により依存関係が解決されてロードされるため、ユーザはどのファイルが必要か詳細に指定する必要なく永続キャッシュに必要なファイルをキャッシュできる効果がある。また、依存解析により必要なファイルのみが永続キャッシュにキ

キャッシュされ、それ以外のファイルは通常キャッシュにキャッシュされるため、永続キャッシュは効率良く利用される効果がある。

【0083】請求項2記載の発明によれば、不揮発性記憶装置中にキャッシュ履歴簿を設けるように構成したので、プログラム実行前だけでなくプログラム実行途中や終了後からでも遡ってプログラム実行開始時点からの特定期間、そのプログラム実行開始時点からのプログラムファイルの永続キャッシュへのキャッシュが可能となるばかりではなく、通常キャッシュにキャッシュされている内容を永続キャッシュにコピーすることにより、通信量の削減が可能となり、さらにその時点でネットワークに接続されていなくとも永続キャッシュの構築が可能となる効果がある。

【0084】請求項3記載の発明によれば、クライアント計算機にプログラムロード・キャッシュ手段を設けるように構成したので、プログラムを実行しなくとも永続キャッシュにプログラムファイルをキャッシュでき、さらにプログラムファイルが複数のファイルにより構成されている場合にすべての機能を実行することなく、もれなくプログラム全体を構成するファイルをロード、キャッシュすることができる効果がある。

【0085】請求項4記載の発明によれば、キャッシュ移動手段をクライアント計算機に設けるように構成したので、永続キャッシュ中のプログラムファイルの削除とそのプログラムファイルの通常キャッシュへのコピーにより、効率よく永続キャッシュを維持することが可能となり、永続キャッシュから削除されたプログラムファイルも通常キャッシュから削除されない限りネットワークに接続することなく使用することができる効果がある。

【0086】請求項5記載の発明によれば、第2のキャッシュ中のファイル群の削除履歴簿を不揮発性記憶装置中に設けるように構成したので、永続キャッシュの削除履歴簿を参照することにより、削除すべき永続キャッシュのヒントをユーザに与えることができ、効率良く永続キャッシュを維持することが可能となる。

【0087】請求項6記載の発明によれば、バス一致化手段をクライアント計算機に設けるように構成したので、複数の事業所等があるため複数の同じプログラムを提供するサーバがある場合に、1つのサーバからではなくどのサーバからでもプログラムの永続キャッシュへのロードが1つ分の永続キャッシュの容量で可能となり、かつどのサーバからでも永続キャッシュの更新が可能となる効果がある。

【0088】請求項7記載の発明によれば、更新チェック手段をクライアント計算機に設けるように構成したので、永続キャッシュの更新情報が容易にかつ効率良く判断でき、またユーザの指示がなければ永続キャッシュは更新されないため、これまで安定して動作しているなどの理由によりユーザの満足しているバージョンのプログ

ラムを永続キャッシュに残したまま、更新された版のプログラムを試使用することができる効果がある。

【0089】請求項8記載の発明によれば、更新チェック・ロード手段をクライアント計算機に設けるように構成したので、プログラムの更新情報が得られるだけでなく、依存関係解析を行うことにより、プログラムが複数のファイル群により構成されている場合でもプログラム全体をもれなく更新することができる効果がある。

【0090】請求項9記載の発明によれば、使用期限チェック手段をクライアント計算機に設けるように構成したので、試用あるいは課金のためなど、プログラムの使用期限がある場合、使用頻度に応じて使用期限が近付いていることを警告し、ユーザに新たな使用期限を持つプログラムへの更新を促すことができる効果がある。

【図面の簡単な説明】

【図1】 この発明の実施の形態1の計算機システムを組み込んだネットワークを示す構成図である。

【図2】 プログラムロード・実行手段の動作を示すフローチャートである。

【図3】 この発明の実施の形態2の計算機システムを組み込んだネットワークを示す構成図である。

【図4】 実施の形態2のキャッシュ履歴簿の記憶内容を示す図である。

【図5】 実施の形態2のプログラムロード・実行手段の動作を示すフローチャートである。

【図6】 実施の形態2のキャッシュコピー手段の動作を示すフローチャートである。

【図7】 この発明の実施の形態3の計算機システムを組み込んだネットワークを示す構成図である。

【図8】 実施の形態3のプログラムロード・キャッシュ手段の動作を示すフローチャートである。

【図9】 この発明の実施の形態4の計算機システムを組み込んだネットワークを示す構成図である。

【図10】 実施の形態4のプログラムロード・実行手段の動作を示すフローチャートである。

【図11】 実施の形態4のキャッシュ移動手段の動作を示すフローチャートである。

【図12】 この発明の実施の形態5の計算機システムを組み込んだネットワークを示す構成図である。

【図13】 各計算機の保存するプログラムファイルの分布を表す図である。

【図14】 実施の形態5のプログラムロード・実行手段の動作を示すフローチャートである。

【図15】 この発明の実施の形態6の計算機システムを組み込んだネットワークを示す構成図である。

【図16】 実施の形態6のプログラムロード・実行手段の動作を示すフローチャートである。

【図17】 この発明の実施の形態7の計算機システムを組み込んだネットワークを示す構成図である。

【図18】 実施の形態7のプログラムロード・実行手

段の動作を示すフローチャートである。

【図19】 この発明の実施の形態8の計算機システムを組み込んだネットワークを示す構成図である。

【図20】 実施の形態8の不揮発性記憶装置中に設けられた使用期限記録簿の記録内容の一例を示す図である。

【図21】 実施の形態8のプログラムロード・実行手段の動作を示すフローチャートである。

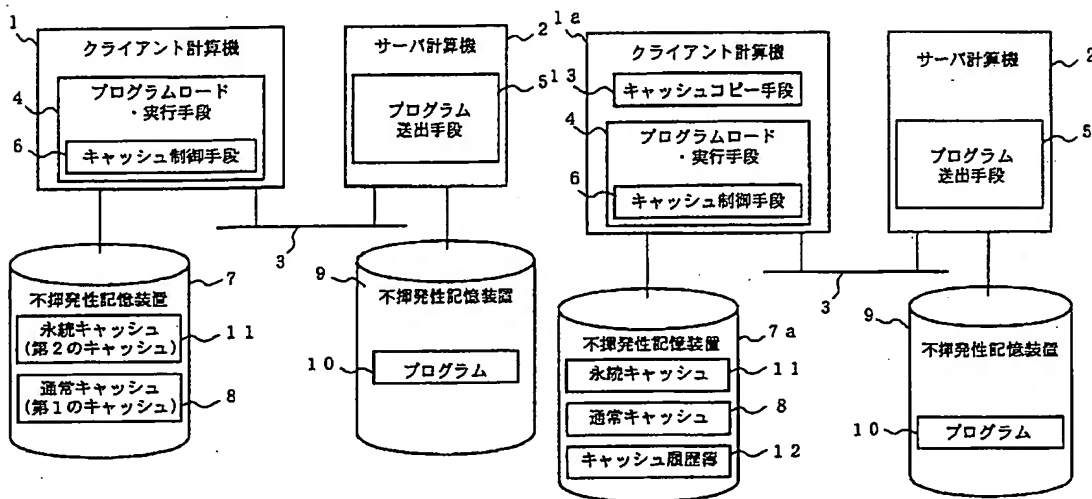
【図22】 従来の計算機システムを組み込んだネットワークの構成図である。

【符号の説明】

1, 1a, 1b, 1c, 1d, 1e, 1f, 1g クライアント計算機、7, 7a, 7b 不揮発性記憶装置、8 通常キャッシュ（第1のキャッシュ）、10 プログラム、11 永続キャッシュ（第2のキャッシュ）、12 キャッシュ履歴簿、14 プログラムロード・キャッシュ手段、15 キャッシュ移動手段、16 パス一致化手段、17 更新チェック手段、18 更新チェック・ロード手段、19 使用期限チェック手段。

【図1】

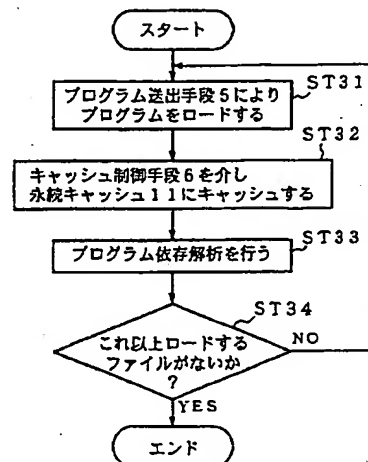
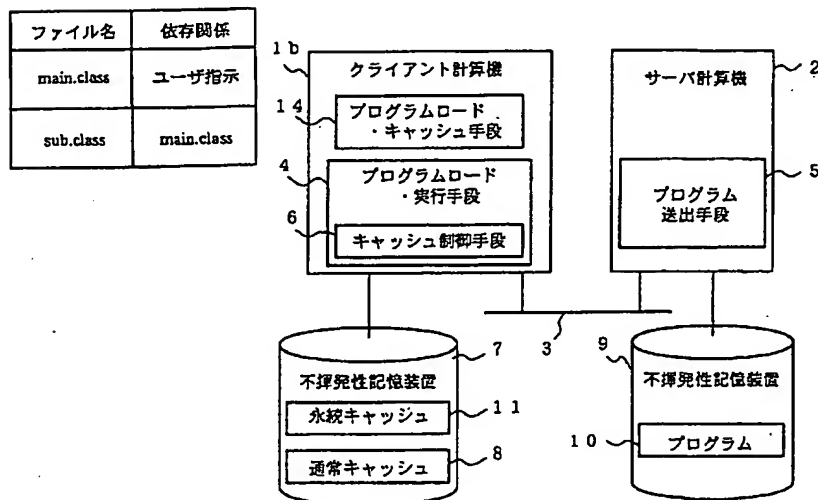
【図3】



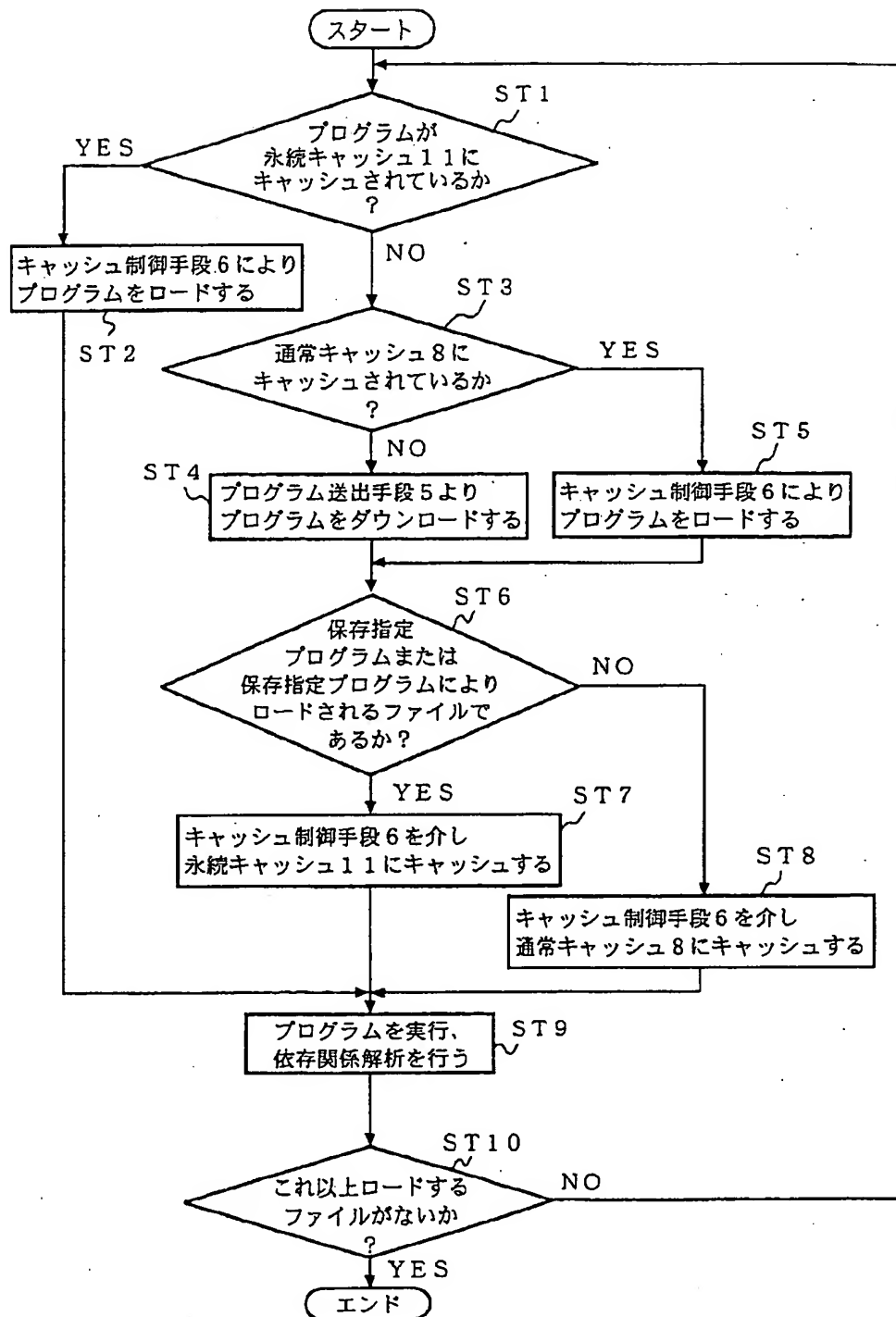
【図4】

【図7】

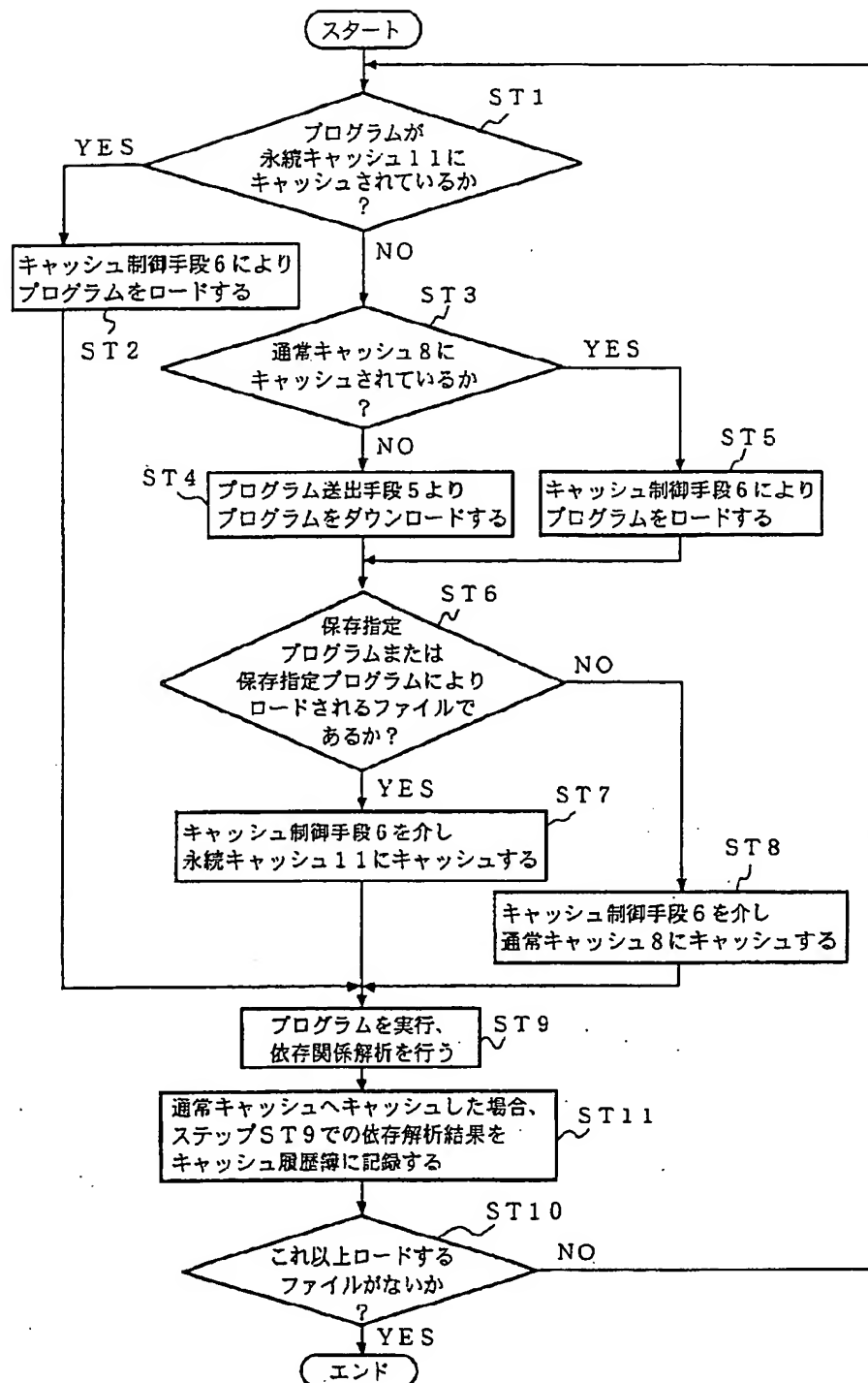
【図8】



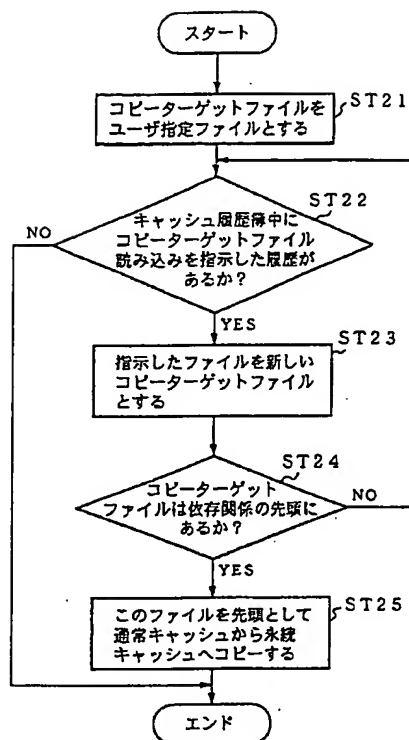
【図2】



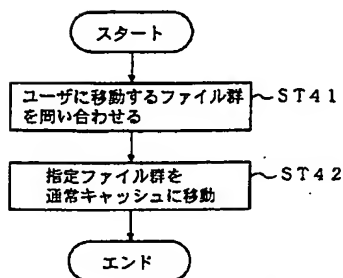
【図5】



【図6】



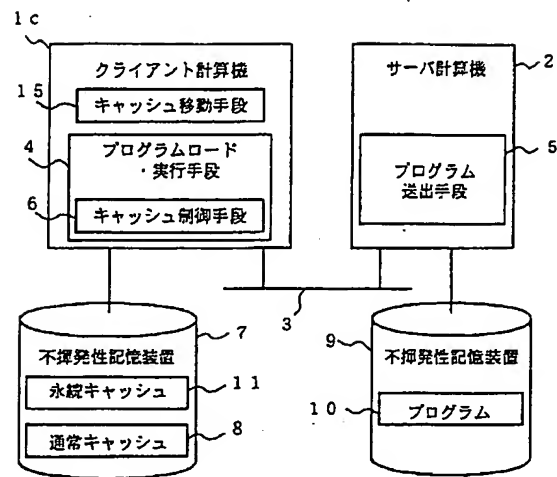
【図11】



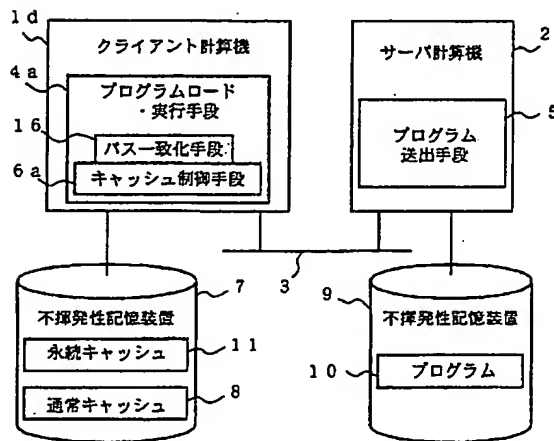
【図20】

ファイル群	使用期限	前回使用
ファイル群A	なし	----
ファイル群B	97/3/10	97/1/25

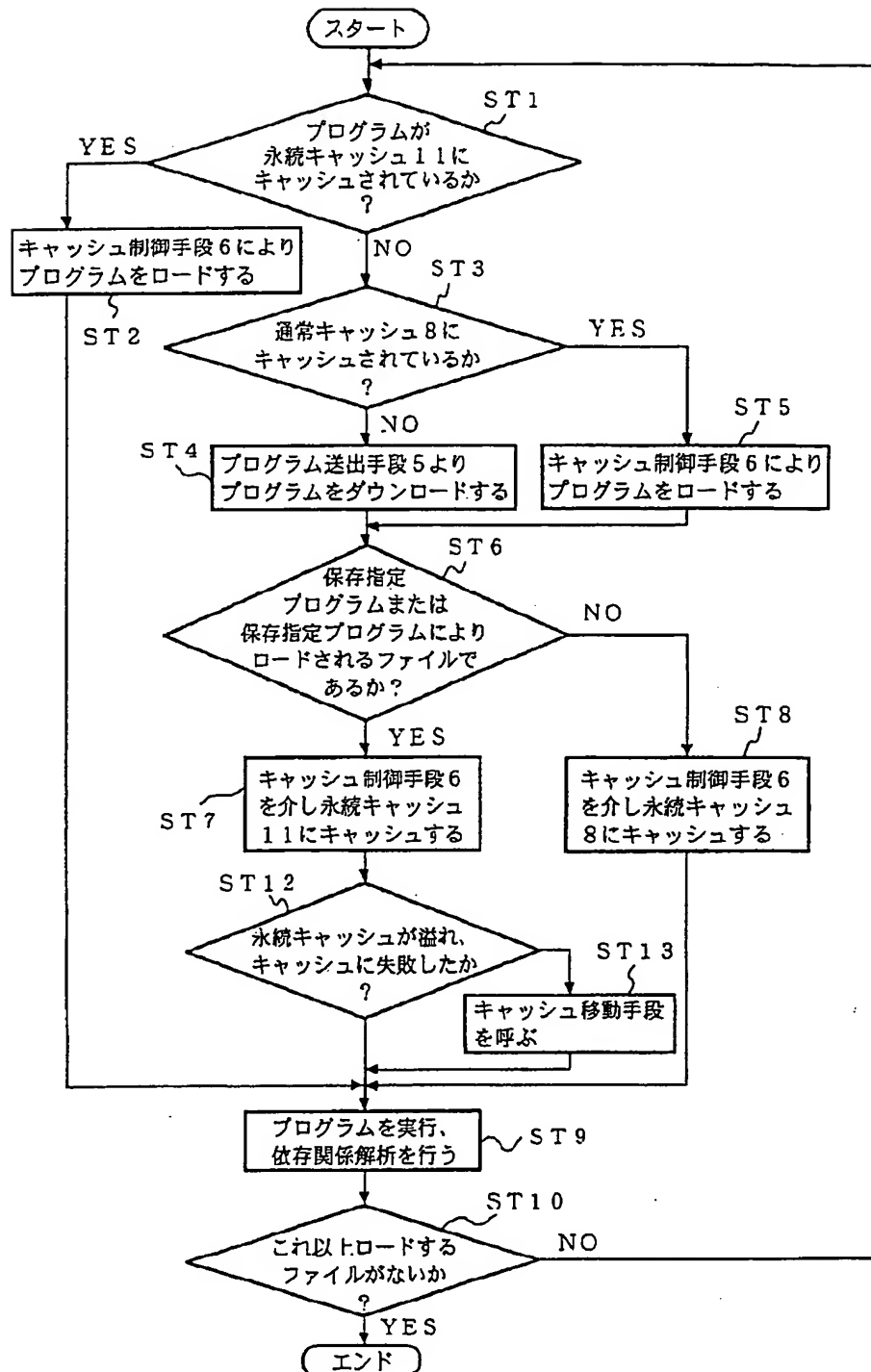
【図9】



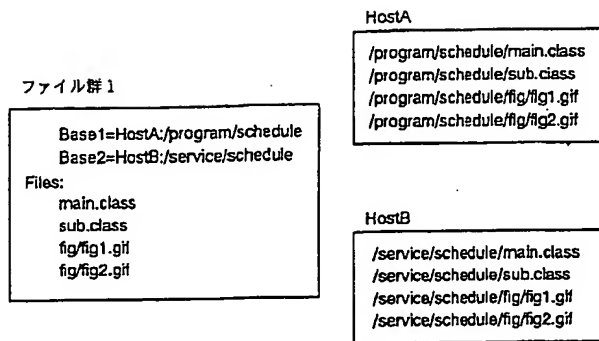
【図12】



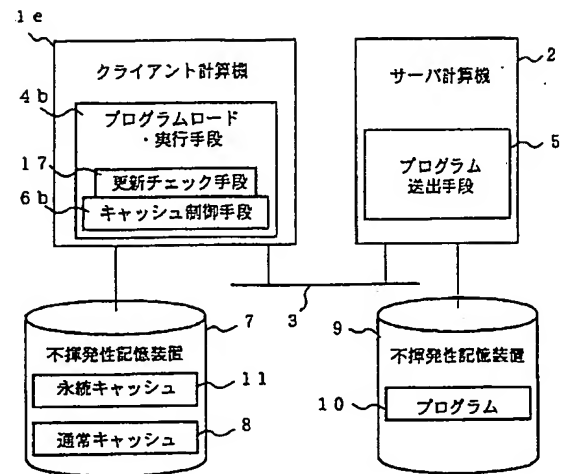
【図10】



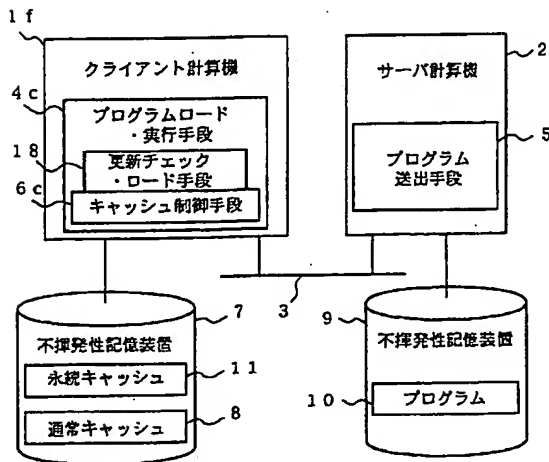
【図13】



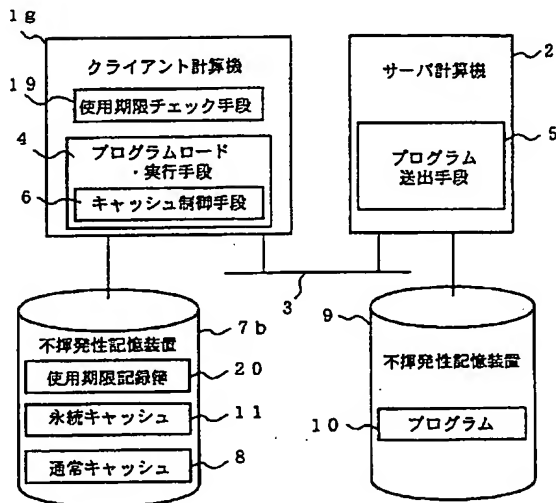
【図15】



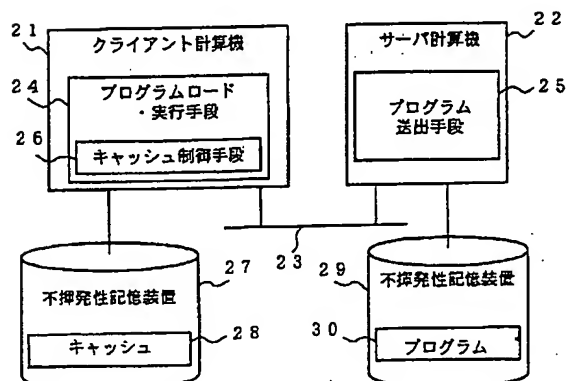
【図17】



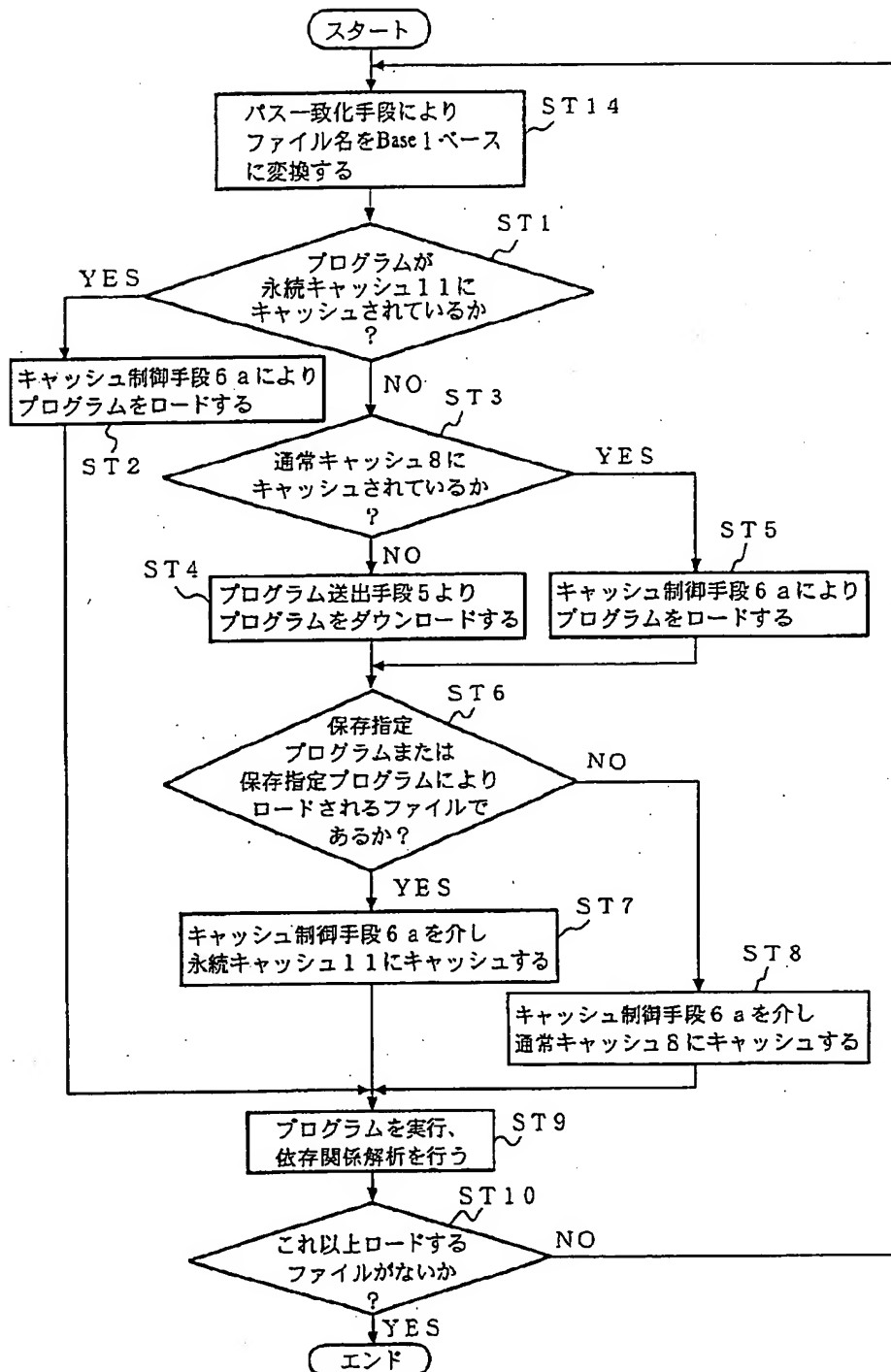
【図19】



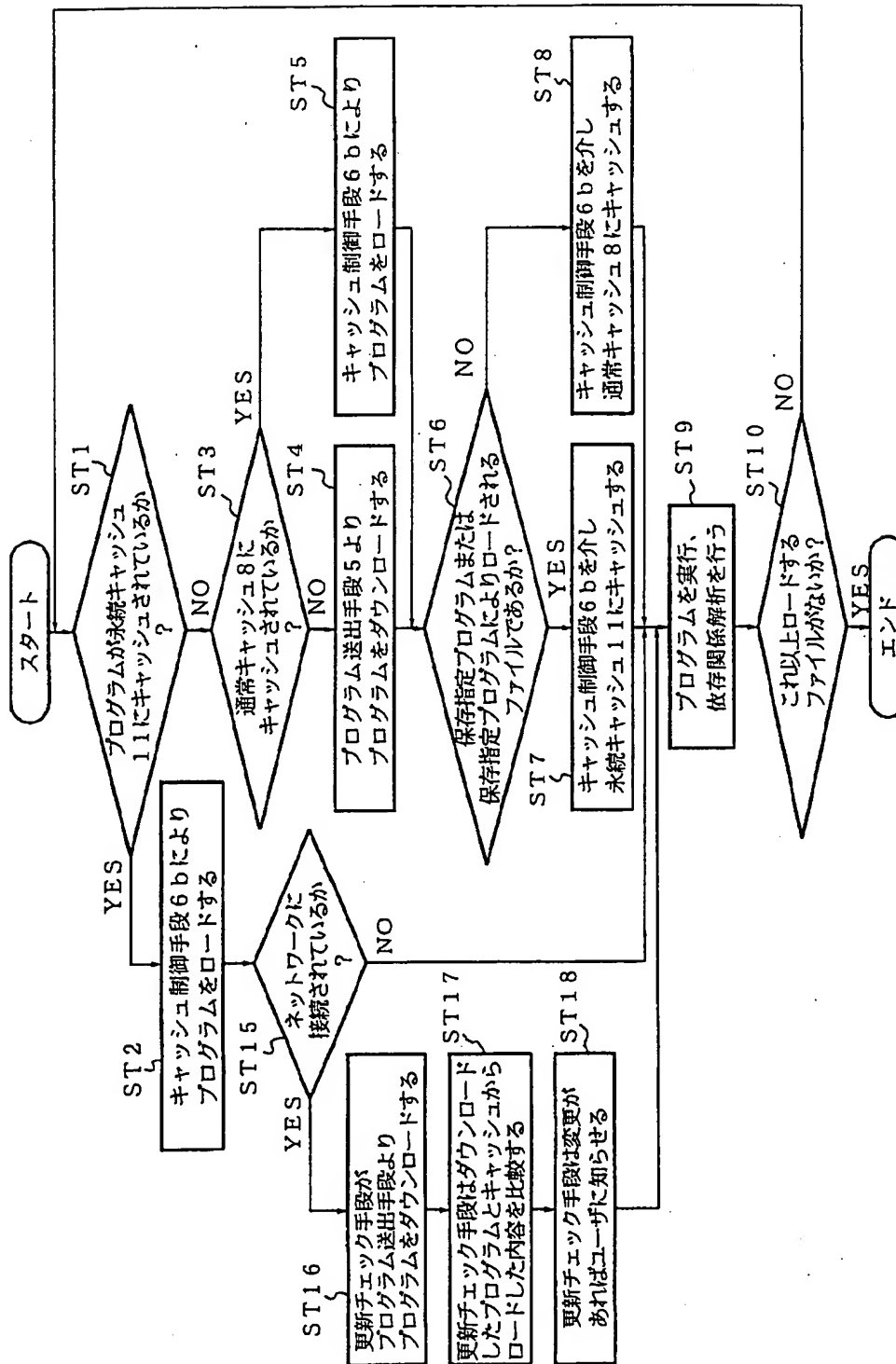
【図22】



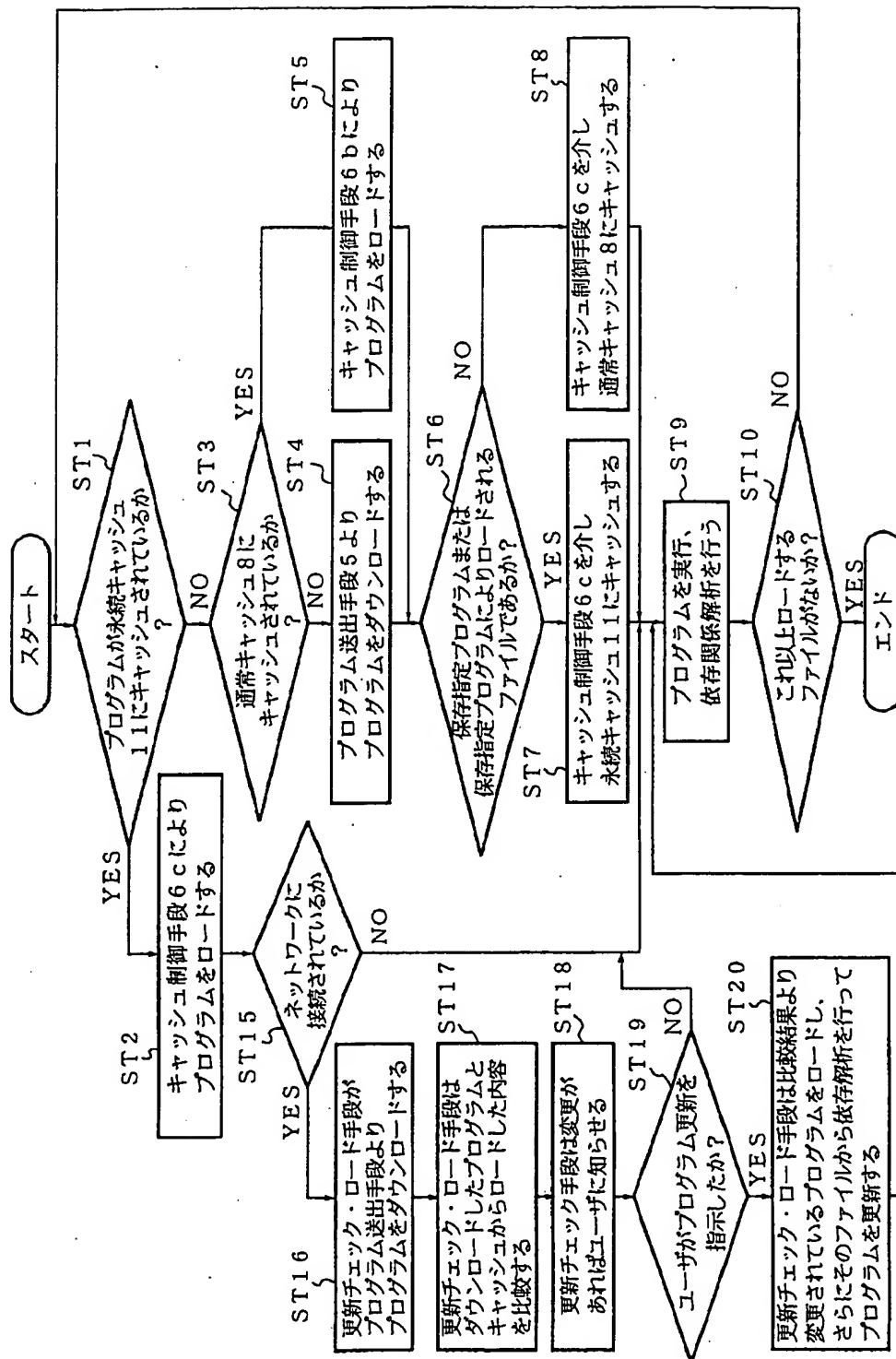
【図14】



【図16】



【図18】



【図21】

